Catalog Description

Course Objectives
A successful student will be able to understand the basic components of a computer operating system, and the interactions among the various components. The course will cover an introduction on the policies for scheduling, deadlocks, memory management, synchronization, system calls, and file systems. The students will implement solutions via C/C++ programs, and through NACHOS.

Prerequisites
- Senior standing or graduate standing in computer science or computer engineering.
- Introduction to C or Introduction to C++ (CIS 506 or CIS 504)
- Discrete mathematics and data structures (CIS 275, CIS 351)
- Java or basic object-oriented programming design principle
- Proficient to, at least, C programming language, basic Unix commands, editors, utilities

Course Outcomes
Upon successful completion of this course, students are expected to have the ability to:

- Describe and explain the fundamental components of a computer operating system. [ABET (a), (i), (j), (k)] Assessment: Students will take midterm exams, final exams, and homework.
- Describe and explain the fundamental components of a computer operating system. [ABET (a), (i), (j), (k)] Assessment: Students will take midterm exams, final exams, and homework.
- Define, restate, discuss, and explain the policies for scheduling, deadlocks, memory management, synchronization, system calls, and file systems. [ABET (a), (i), (j), (k)] Assessment: Students will take midterm exams, final exams, and homework.
- Describe and extrapolate the interactions among the various components of computing systems. [ABET (a), (i), (j), (k)] Assessment: Students will take midterm exams, final exams, and homework.
- Design and construct the following OS components: System calls, Schedulers, Memory management systems, Virtual Memory and Paging systems. [ABET (a), (c), (i), (j), (k)] Assessment: Students will design and implement the above OS components within NACHOS with C++.
- Illustrate, construct, compose and design solutions via C/C++ programs, and through NACHOS. [ABET (a), (c), (i), (j), (k)] Assessment: Students will design and implement the above OS components within NACHOS.
• Measure, evaluate, and compare OS components through instrumentation for performance analysis. [ABET (b), (j)] Assessments: (1) Students will run experiments on their own implemented OS components and the components provided by NACHOS and (2) Students will perform scientific analysis on the performance of the components and are asked to submit a short paper on their experimental results.

• Discuss with fellow students about designing new components of OS. [ABET (a), (b), (d), (c), (j), (k)] Assessment: There will be one hour of discussion session in the semester to discuss this matter. The session is to provoke alternative design ideas for OS components. The exams and homework will assess the outcomes.

Outcome Measurement
There are about three programming projects in the course. The first one is usually C/C++ review project with dynamic memory allocation, linked lists, etc. (two weeks). The second one is synchronization and mutual exclusion programming using Nachos. The students are required to solve at least three classical synchronization problems using semaphores. The second project is due in three weeks usually. The third project consists of two different parts: (1) Implementation of preemptive round-robin time-sharing scheduling, and (2) Implementations of various system calls. In total, students have about a month to finish the project for both two parts combined.

Course Topics
Processes and threads, Deadlocks, Memory management, System calls, I/O, File systems, Unix and Linux (Nachos), Windows 2000

CAC Category Content

1 Data Structures
0.5 Algorithms
1 Software Design
1 Computer Organization & Architecture
0.5 Programming Languages